

**IN THE SPECIFICATION**

**Please amend the paragraph beginning on page 3, line 11, as follows:**

Figure 9 shows one embodiment of processor 900. Each processor 110 is composed of a scalar processor 910, ~~and~~ two vector pipes 930 and a Translation Look-aside Buffer (TLB) 940. The scalar and vector unit are decoupled with respect to instruction execution and memory accesses. Decoupling with respect to instruction execution means the scalar unit can run ahead of the vector unit to resolve control flow issues and execute address arithmetic. Decoupling with respect to memory accesses means both scalar and vector loads are issued as soon as possible after instruction dispatch. Instructions that depend upon load values are dispatched to queues where they await the arrival of the load data. Store addresses are computed early and their addresses saved for later use. Each scalar processor 910 is capable of decoding and dispatching one vector instruction (and accompanying scalar operand) per cycle. Instructions are sent in order to the vector units, and any necessary scalar operands are sent later after the vector instructions have flowed through the scalar unit's integer or floating point pipeline and read the specified registers. Vector instructions are not sent speculatively; that is, the flow control and any previous trap conditions are resolved before sending the instructions to the vector unit. For a further description of decoupled vector architecture please refer to the U.S. patent application entitled "Decoupled Vector Architecture", filed on even date herewith, the description of which is hereby incorporated by reference.

**Please amend the paragraph beginning at page 6, line 19, as follows:**

Figure 10 shows one embodiment of local memory 1000 used in the processing node 500 of Figure 5. In this embodiment, local memory includes two MSP ports 1010, two Cache Coherence Directories 1040, a crossbar switch 1020, two network ports 1030, a Remote Address Translation Table (RTT) 1050, and RAM 1060. Remote Translation Table (RTT) 1050 translates addresses originating at remote processing nodes 500, 600, 700, 800 to physical addresses at the local node. In some embodiments, this includes providing a virtual memory address at a source node, determining that the virtual memory address is to be sent to a remote node, sending the virtual memory address to the remote node, and translating the virtual memory

address on the remote node into a physical memory address using a RTT. The RTT contains translation information for an entire virtual memory address space associated with the remote node. Another embodiment of RTT provides for translating a virtual memory address in a multi-node system. The method includes providing a virtual memory address on a local node by using a virtual address of a load or a store instruction, identifying a virtual node associated with the virtual memory address, and determining if the virtual node corresponds to the local node. If, instead, the virtual node corresponds to a remote node, then the method includes sending the virtual memory address to the remote node, and translating the virtual memory address into a physical memory address on the remote node.

Fig. 11 illustrates a format for a virtual memory address, according to one embodiment. In this embodiment, virtual memory address format 1100 contains a 64-bit virtual address space. Bits 37..0 represent a virtual offset into virtual memory space, wherein potential page boundaries range from 64 KB to 4 GB. Bits 47..38 represent the Vnode (i.e., virtual node). This is used by the hardware when performing remote address translation. Bits 61..48 must be set to zero in this implementation. Bits 63..62 specify the memory region, which determines the type of address translation used in kernel mode. The virtual address space can be considered a flat virtual address space for uniprocessor, or symmetric multiprocessing applications. As stated, this embodiment supports eight page sizes ranging from 64 KB to 4 GB. Thus, the page boundary can vary, from between bits 15 and 16, to between bits 31 and 32.

In various embodiments of the invention, virtual addresses used for instruction fetches and data references are first translated into physical addresses before memory is accessed. These embodiments support two forms of address translation: source translation, and remote translation. The first form of address translation is source translation, in which a virtual address is fully translated by a Translation Look-aside Buffer (TLB) on a local P chip to a physical address on an arbitrary node. The second form of address translation is remote translation, in which the physical node number is determined by a simple translation of the virtual address Vnode field, and the remaining virtual address VOffset field is sent to the remote node to be translated into a physical address offset via a Remote-Translation Table (RTT). The type of address translation performed is based upon values in a configuration control register and the virtual address itself. Remote translation is performed if all of the following three conditions are

true: (1) Remote translation is enabled (e.g., a flag contained in the configuration control register is set); (2) The virtual address is to the user region (Bits 63..62 = 00 in the virtual address); and (3) The virtual address references a remote node (Bits 47..38 in the virtual address are not equal to a local node value contained in the configuration control register). If any of the above conditions are false, then source translation is performed. Remote translation can be enabled/disabled on a per-processor basis.

Fig. 12 illustrates a flow diagram for analyzing a VNode field in the virtual memory address, according to one embodiment of the present invention. Flow diagram 1200 includes blocks 402, 406, and 408, and also includes checkpoint 404. Flow diagram 1200 illustrates one way in which a virtual memory address can be translated into a physical memory address (in either local or remote memory space). Block 402 includes identifying the virtual node from a virtual address. In one implementation, a local node can identify the virtual node by looking at the VNode field of the virtual address. Checkpoint 404 determines if the virtual node is the same as, or equal to, the local node. If so, flow diagram 1200 continues to block 406, wherein the virtual address is translated into a physical address locally using a Translation Look-Aside Buffer (TLB). The local node is then able to address local physical memory space. If the virtual node is not the same as the local node, then flow diagram 1200 continues to block 408, wherein the virtual address is translated into a physical address remotely (on a remote node) using a Remote-Translation Table (RTT). In this fashion, the local node is effectively able to address remote memory space of the remote node.

For a further description of RTTs please refer to the U.S. patent application entitled “Remote Translation Mechanism for a Multi-node System”, U.S. Application No. 10/235,898, filed September 4, 2002; “Remote Translation Mechanism for a Multinode System”, filed on even date herewith, and “Method for Sharing a Memory within an Application Using Scalable Hardware Resources”, filed on even date herewith, the description of which are hereby incorporated by reference.